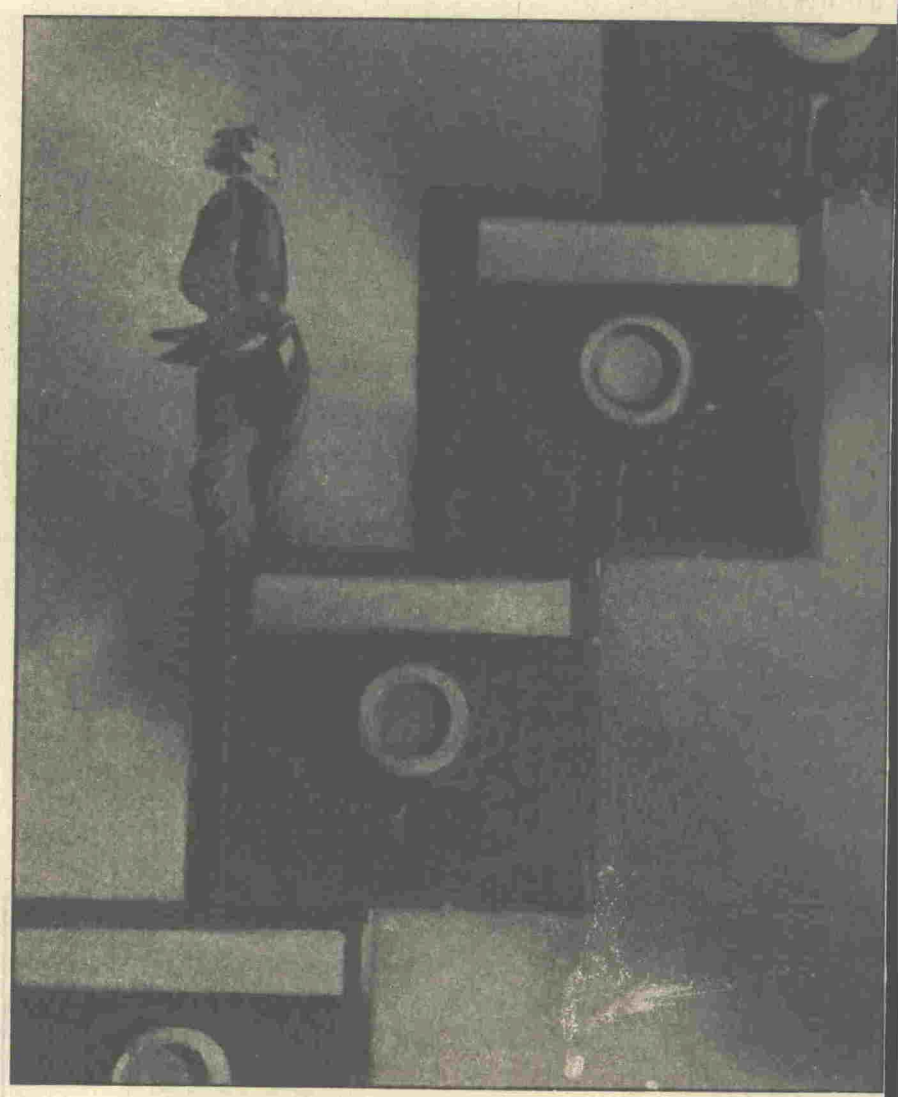
อาจหาญ สัตยารักษ์

SOFT MAIL

ทางเลือกใหม่ในการจัดการฐานข้อมูลดีเบส (2)

เขียนปาสคาลในการแก้ไข ค้นหา เปลี่ยนแปลงข้อมูลในไฟล์ดีเบส



เป็นไงบ้าง...สำหรับตอนที่ 1 ที่ผ่านไปนั้น ผู้อ่านคงได้ประโยชน์ในการทำงานด้านฐานข้อมูลของท่านไม่มากก็น้อย มาถึงวันนี้คิดว่าคงจะมีผู้รู้หลายท่านได้เขียนชุดคำสั่งเพิ่มเติมเพื่อจัดการในเรื่องอื่น ๆ เสร็จไปเรียบร้อยแล้วก็ได้ แต่หากยังไม่ได้ทำก็ลองดูแนวคิดที่ผู้เขียนจะเสนอในตอนที่ 2 นี้ต่อไป

สำหรับตอนที่ 2 นี้จะเขียนถึงเรื่องการแก้ไขข้อมูล, การค้นหาข้อมูล, การเปลี่ยนแปลงข้อมูล ฯลฯ เพื่อเป็นแนวทางในการใช้ปาสคาลจัดการกับฐานข้อมูลดีเบสของท่านให้มีความสมบูรณ์ยิ่ง ๆ ขึ้น

คราวนี้ตัวโปรแกรมอาจจะยาวเล็กน้อย โดยในซอฟต์เมล์จะอยู่ในไฟล์ชื่อ dbase 2. pas โดยมีแฟ้มข้อมูลตัวอย่างชื่อ dbase2.dbf ไว้ให้ทดลองดำเนินการ

ตัวแปรที่เพิ่มขึ้น

เพื่อให้การทำงานในตอนที่สองนี้มีความสมบูรณ์ยิ่งกว่าที่ได้กล่าวมาในตอนแรก จึงมีความจำเป็นต้องเพิ่มตัวแปรที่ส่วนหัวอีกกลุ่มหนึ่ง ดังนี้

```
Numeric, Edit, UseDbf:boolean;
NoField:byte; Xstart:word;
NumField:real; InputSTR:string;
Funckey:char;
FieldSTR:array (1..30) of string;
Space:string (40);
```

เพื่อใช้งานดังต่อไปนี้

Numeric สำหรับการป้อนข้อมูลในกรณีที่ชนิดของเขตข้อมูลเป็นตัวเลข (N)

Edit สำหรับตรวจว่าเป็นการป้อนข้อความในช่วงการแก้ไขข้อมูลหรือไม่

UseDbf สำหรับตรวจว่ามีแฟ้มข้อมูลดีเบสอยู่ในหน่วยความจำหรือไม่

NoField ลำดับที่ของเขตข้อมูล โดยมีตัวแปร Xstart เป็นตัวกำหนด

ดเริ่มต้นของเขตข้อมูล

NumField สำหรับเก็บค่าเป็นตัว าขของข้อมูลที่ถูกบรรจุในเขตข้อมูล นิดตัวเลข

InputSTR ตัวแปรเก็บข้อความ ป้อนผ่านเป็นพิมพ์จากชุดคำสั่ง Input TR ที่เขียนขึ้นใหม่

Funckey สำหรับเก็บค่าของ งก์ชันคีย์เพื่อใช้ในการเลื่อนแถบสว่าง นเมนูหลัก

FieldSTR เพื่อเก็บข้อมูลที่บันทึก ว้ในเขตข้อมูลนั้น ๆ เพื่อนำไปใช้ใน อนแก้ไขข้อมูล

Space เป็นตัวแปรเก็บข้อความที่ ระกอบด้วยช่องว่าง 40 ตัว

รับชุดคำสั่งเดิม

สำหรับตอนที่ 2 นี้เราจะปรับชุด าสั่งในตอนที่ 1 ใหม่ดังนี้

ชุดคำสั่ง ReadStructure ให้ตัด วนการอ่านข้อมูลจากแฟ้มฐานข้อมูล งแต่ assign ไปจนถึง close ออกไป งหมดรวม 6 บรรทัด

ตัดชุดคำสั่ง ReadDatabase อกไป แล้วมาใช้ชุดคำสั่ง ReadData-ase ของตอนนี้แทน

เปลี่ยนชุดคำสั่ง DisplayDatabase, ppendDatabase ของเดิมมาใช้ที่เขียน หม่ในตอนนี้แทน เพราะจะทำงานได้ มบูรณ์กว่าของเดิม

ให้ตัดโปรแกรมหลักจากรูปที่ 8 อกไปทั้งหมด

ดคำสั่งช่วยงานเพิ่มเติม

ก่อนจะเขียนถึงการจัดการกับฐาน อมูลดีเบสในเรื่องอื่น ๆ ต่อไปนั้น ผู้ ียนขอแทรกการเขียนชุดคำสั่งที่จำเป็น บางอย่างเสียก่อน ทั้งนี้เพื่อให้การทำงาน สมบูรณ์ยิ่งขึ้น โดยมีชุดคำสั่งช่วยงาน ดังนี้

ชุดคำสั่ง Readchr ชุดคำสั่งใน การตรวจการกดแป้นทั้งแป้นปกติและ ฟังก์ชันคีย์

ชุดคำสั่ง Inverse แสดงข้อความ แบบกลับขาวดำ

ชุดคำสั่ง Normal แสดงข้อความ แบบปกติ

ทั้งสามชุดคำสั่งนี้มาจากรูปที่ 1 หน้า 285 ฉบับที่ 54 เรื่อง การจัดการไฟล์ด้วยเทอร์โบปาสคาล โดยให้ผู้อ่านพิมพ์ชุดคำสั่งดังกล่าวทั้งสามนี้ไว้ที่ต้น ๆ ไฟล์ด้วย

ชุดคำสั่งอีก 2 ชุด คือการจัดการกับเคอร์เซอร์เป็นดังรูปที่ 1 โดยมี 2 ชุดคำสั่งดังนี้

ชุดคำสั่ง CursorOff ในรูปที่ 1 นั้น มีไว้ใช้ตอนเลื่อนแถบสว่างในการเลือกเมนู (จะกล่าวถึงในภายหลัง) ซึ่งในช่วงนั้นมีความประสงค์ไม่ให้มีแสดงเคอร์เซอร์ ส่วนชุดคำสั่ง CursorOn ก็ทำงานกลับกันคือจะแสดงเคอร์เซอร์ตามปกติ

ส่วนชุดคำสั่งในรูปที่ 2 นั้นเป็นการป้อนข้อความเพื่อให้ได้รูปแบบตามวิธีการของดีเบส จึงขอให้ทำความเข้าใจกับชุดคำสั่งเหล่านี้ก่อน

ที่นี้เราต้องมาทำความเข้าใจกับชุดคำสั่งในรูปที่ 2 ซึ่งถือว่ามีความสำคัญ เพราะจะช่วยให้เราป้อนข้อความได้ตามขนาดของความกว้างของแต่ละเขตข้อมูลของฐานข้อมูลที่เราเรียกใช้ได้ โดยจะนำมาแทนคำสั่ง readln เดิมที่ใช้อยู่ เพื่อให้การป้อนข้อความตรงกับความต้องการที่กำหนดไว้ในเขตข้อมูลนั้น

โดยขออธิบายชุดคำสั่ง InputSTR คร่าว ๆ ดังนี้

ตัวแปรที่ส่งเข้ามานั้นจะเป็นดังนี้

x,y คือ สดมภ์และแถวที่ต้องการป้อนข้อความ

count จำนวนความกว้างของเขตข้อมูลที่จะป้อนหรือจำนวนตัวอักขระทั้งหมดที่ต้องการป้อน

Decimal จำนวนเลขหลังจุดทศนิยม (จะมีค่าเป็นศูนย์ถ้าชนิดของเขตข้อมูลเป็นข้อความ)

NoF หมายเลขของเขตข้อมูลเพื่อนำมาหาชนิดของเขตข้อมูลนั้น

```
PROCEDURE CursorOff;
var  Vport :integer Absolute $0040:$0063;
     CursorMode :integer Absolute $0040:$0060;
begin
    port[Vport]:=10;
    port[Vport+1]:=Hi(CursorMode) or $20;
    port[Vport]:=11;
    port[Vport+1]:=Lo(CursorMode);
end;

PROCEDURE CursorOn;
var  Vport :integer Absolute $0040:$0063;
     CursorMode :integer Absolute $0040:$0060;
begin
    port[Vport]:=10;
    port[Vport+1]:=12;
    port[Vport]:=11;
    port[Vport+1]:=13;
    CursorMode :=(12 shl 8) or 13;
end;
```

รูปที่ 1 ชุดคำสั่งจัดการเกี่ยวกับเคอร์เซอร์

```
PROCEDURE Input_STR(x,y,count,Decimal,NoF:byte);
var No,n,p,DotP:byte; ZeRo:string[40];
begin
  fillchar(Zero,41,'0'); Zero[0]:=#40;  n:=0;  DotP:=Count-Decimal;
  Inverse(x,y,copy(space,1,count));     InputSTR:='';
  if Decimal<>0 then Inverse(x+Count-Decimal-1,y,'.');
 Repeat
    gotoxy(x+n,y);  readchr;
    if TypeOfField[NoF]='N' then
        if ch in [#32..#44,#47,#58..#255] then ch:=#0;
    if (ch='.') and (decimal<>0) then begin
       if n<>DotP then begin    if InputSTR='' then InputSTR:='0';
        InputSTR:=copy(space,1,DotP-1-length(InputSTR))+InputSTR;
        InputSTR:=InputSTR+Ch;   n:=count-decimal;
        Inverse(x,y,InputSTR); ch:=#0;
                        end;              end;
  case ch of
   #32..#255:begin  if n<Count then                 begin
                      InputSTR:=InputSTR+Ch;       Inc(n);
                    if (Decimal<>0) and (n=DotP-1) then      begin
                        InputSTR:=InputSTR+'.'; Inc(n);  end;
                      Inverse(x,y,InputSTR)       end
                    Else begin Sound(150); delay(400); nosound; end;
               end;
   #8:begin if n>0 then  begin
      If (TypeOfField[NoF]='N') and (InputSTR[length(InputSTR)]='.') then
          begin  InputSTR[0]:=chr(ord(InputSTR[0])-1); dec(n);  end;
         InputSTR[0]:=chr(ord(InputSTR[0])-1); dec(n);
         Inverse(x,y,InputSTR+' ');
      end;                end;
  end;
until ch=#13;
if (InputSTR='') and Edit then  begin InputSTR:=FieldSTR[NoF];
        Inverse(x,y,InputSTR);  end;
if TypeOfField[NoF]='N' then
  begin
   if Decimal=0 then                                          begin
      if (Count=1) and (InputSTR='') then InputSTR:='0';  end;
   if (Decimal<>0) and (length(InputSTR)<>Count) then begin
      if InputSTR='' then  InputSTR:='0.0'
      else begin
         if DotP>length(InputSTR) then
           InputSTR:=copy(Space,1,DotP-Length(InputSTR)-1)+InputSTR+'.';
              p:=pos('.',InputSTR);
         if p<>0 then begin
            if p<=length(InputSTR) then
               InputSTR:=InputSTR+copy(Zero,1,Count-length(InputSTR));
                      end
       else
        InputSTR:=InputSTR+'.'+copy(Zero,1,Count-length(InputSTR)-1);
          end;                                              end;
        p:=pos('.',InputSTR);
   if p<>DotP then
     InputSTR:=Copy(copy(space,1,Count),1,Count-length(InputSTR))+InputSTR;
     InputSTR[0]:=chr(count);   Inverse(x,y,InputSTR);
 end;
end;
```

รูปที่ 2 ชุดคำสั่งสำหรับป้อนข้อความตามรูปแบบของดีเบส

การป้อนผ่านแป้นพิมพ์ในกรณีที่ชนิดของเขตข้อมูลเป็นตัวเลข ป้อนได้เฉพาะตัวเลขและ . – เท่านั้น ถ้าการกดแป้นพิมพ์เมื่อค่า n<count ก็ให้ทำการเพิ่มตัวแปร InputSTR ด้วยอักขระที่ป้อนเข้ามา โดยเพิ่มค่าตัวแปร n ขึ้นทีละ 1

ถ้าเขตข้อมูลที่ป้อนเป็นชนิดตัวเลขและได้ป้อนมาถึงตำแหน่งจุดทศนิยมแล้วก็ให้ทำการบวก . เข้าไปในตัวแปรและเพิ่มค่า n อีก 1 หรือป้อน . เข้ามาก็จะตรวจการป้อนให้ตรงกับความจริงทันที

หากการป้อนนั้นเกิดเมื่อค่า n>=count ก็จะมีเสียงเตือนให้ทราบและจะไม่รับรู้การกดแป้นนั้น ๆ ยกเว้นการกดแป้น Enter ซึ่งถือเป็นการยุติการป้อน (ถ้าไม่ชอบใจเสียงที่กำหนดมาก็ให้เปลี่ยนค่าตัวเลขได้)

แต่ในกรณีที่กดแป้นลบอักขร
(#8) นั้นก็จะทำงานในทางกลับกัน คื
จะลดตัวแปร n ลงและตัดอักขระตั
ท้ายสุดของตัวแปร InputSTR ออกไ
และถ้าพบ ในกรณีที่ชนิดของเข
ข้อมูลเป็นตัวเลขก็จะขยับข้ามไปยังตำ
เลขข้างหน้าเลย

หลังจากกดแป้น Enter ถ้าพบว่
อยู่ในช่วงการแก้ไขข้อมูล (edit:=true
ก็ให้ตัวแปรที่ป้อนมีค่าเท่ากับข้อมูลใ
เขตข้อมูลเดิม คือ InputSTR:=Fiel
STR[NoF];

หลังจากกดแป้น Enter แล้ว ถ้
ชนิดของเขตข้อมูลเป็นตัวเลข ก็จะม
ทำการตรวจข้อความที่ป้อน เพื่อให้ไ
รูปแบบตรงกับที่ดีเบสกำหนดไว้ เช่
จัดให้เป็น 0.0, 12.0 เป็นต้น เมื่อไ
จัดการตามรูปแบบของดีเบสแล้วก็จ
แสดงข้อความที่เปลี่ยนแล้วนี้บนจอภา
อีกครั้ง

การแสดงข้อมูลและการอ่านแฟ้มข้อมูล

จากตอนที่ 1 นั้นเป็นส่วนเริ่มต้
การศึกษา ผู้เขียนชุดคำสั่งไว้ไม่สมบูร
นัก เพราะยังขาดการตรวจว่ามีแฟ้
ตามชื่อที่ระบุให้เรียกใช้นั้นอยู่ในแผ่
บันทึกหรือไม่ ส่วนการแสดงข้อมูลนั้
ยังไม่สามารถนำไปใช้ในกรณีที่จะต้อง
การแก้ไขข้อมูลได้ ยังต้องปรับปรุงชุ
คำสั่ง DisplayDatabase และเขีย
ReadDatabase ขึ้นใหม่ เพื่อใช้งานไ
สมบูรณ์ต่อไปและจะนำไปใช้ในตอ
ต่อ ๆ ไปด้วย

ตอนนี้จึงขอให้ผู้อ่านศึกษาชุดคำส
ทั้งสองในรูปที่ 3 ให้เข้าใจก่อน

จากชุดคำสั่ง DisplayDataba

นั้นมีสิ่งที่เปลี่ยนไปคือ มีการนำข้อมูล
ที่บรรจุในแต่ละเขตข้อมูลของระเบียน
ข้อมูลที่กำลังอ่านมาใส่ในตัวแปร Field
STR[No] เพื่อนำไปใช้งานในช่วงแก้ไข
ข้อมูล

ชุดคำสั่ง Read-Database นั้นคง
จะทำความเข้าใจไม่ยากนักเพราะได้ปรับ
ปลี่ยนมาจากชุดคำสั่ง Readstructure
โดยตัดส่วนการแสดงผลบนจอภาพ
ออกไป และเพิ่มการตรวจชื่อแฟ้มฐาน
ข้อมูลเพื่อป้องกันความผิดพลาดในกรณี
ไม่มีแฟ้มนั้นอยู่ในแผ่นบันทึกที่กำหนด
ให้อ่านแฟ้มข้อมูล

การแก้ไขข้อมูล

การแก้ไขข้อมูลที่บรรจุในระเบียน
ข้อมูลใด ให้เรียกใช้ชุดคำสั่งในรูปที่ 4
คือ EditDatabase ซึ่งจะทำงานดังนี้

แสดงข้อมูลเดิมที่มีอยู่ในระเบียน
ข้อมูลนั้น ๆ (R) ลงบนจอภาพ

ทำการแก้ไขข้อมูลเรียงลำดับจาก
เขตข้อมูลแรกไปจนถึงเขตข้อมูลสุดท้าย
โดยการเรียกใช้ชุดคำสั่ง InputSTR ซึ่ง
จะยให้การป้อนข้อมูลตรงกับรูปแบบของ
ข้อมูลที่กำหนดในเขตข้อมูลนั้น ๆ เพราะ
ได้ส่งตัวแปร คือ widthOfField[no],
Dec (no) และ No ไปให้ตรวจสอบการ
ป้อนข้อมูลของเขตข้อมูลนั้น ๆ

จากนั้นนำข้อมูลที่แก้ไขแล้วนั้น
ไปเก็บแทนที่ข้อมูลเดิมตามคำสั่งช่วง
รอบ for ตัวแปร s

เมื่อได้แก้ไขครบทุกเขตข้อมูลแล้ว
จะมีคำถามว่าจะทำการแก้ไขระเบียน
ข้อมูลอื่น ๆ อีกหรือไม่ ถ้าต้องการให้กด
คีย์ Y ก็จะทำการแก้ไขระเบียนข้อมูล
ชุดถัดไป ถ้าไม่ต้องการก็จบส่วนการ
แก้ไขข้อมูล

ขอให้ศึกษาชุดคำสั่งที่ปรากฏในรูป

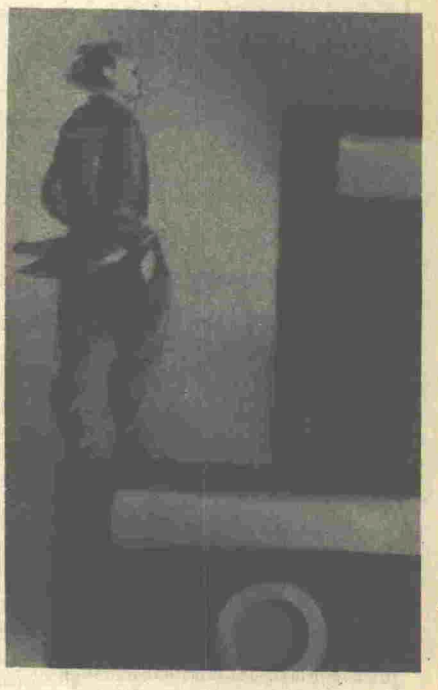
```
PROCEDURE DisplayDatabase(R:word);
var no,s,x,xx:word;
begin
   xx:=1;  writeln('========== Record no.',R,' ============');
 for No:=1 to CountOfField do
   begin
       write('  ',NameOfField[no],': ');
     if R<=CountOfRecord then
       begin  FieldSTR[No]:='';
        x:=(R-1)*SizeOfRecord+xx;
        for s:=x to (x+WidthOfField[No]-1) do
                FieldSTR[No]:=FieldSTR[No]+DataDbase[s];
       end;
        writeln(FieldSTR[No]);   xx:=xx+WidthOfField[no];
   end;
end;

PROCEDURE Read_DataBase;
var no:word; k:byte;
begin
   assign(f,Dbasefile);
   {$I-}  reset(f,1); {$I+}
 if IOResult<>0 then  begin  UseDbf:=false;
    gotoxy(20,21); write('FILE :',DbaseFile,' not found ');
    readchr;          end
 else     Begin
   blockread(f,head,10);  SizeOfHead:=Head[8]+Head[9]*256;
   reset(f,1);            blockread(f,head,SizeOfHead);
   CountOfRecord:=Head[4]+Head[5]*256;
   SizeOfRecord:=Head[10]+Head[11]*256;
   blockread(f,DataDbase,SizeOfRecord*CountOfRecord);
   close(f);              CountOfField:=round(SizeOfHead/32)-1;
 for no:=1 to CountOfField  do
  begin
     NameOfField[no]:=copy(Space,1,10);  Fdec[no]:=0;
     TypeOfField[no]:=chr(Head[32*no+11]);
     WidthOfField[no]:=Head[32*no+16];
     k:=0;
      repeat
         NameOfField[no][k+1]:=chr(Head[32*no+k]);
         inc(k);   if head[32*no+k]=0 then k:=27;
      until k>9;
        if TypeOfField[no]='N' then  FDec[No]:=Head[32*no+17];
   end;    end;
end;
```

รูปที่ 3 ชุดคำสั่งในการอ่านแฟ้มข้อมูลและการแสดงข้อมูล

ที่ 4 เปรียบเทียบดู

สำหรับชุดคำสั่ง AppendDatabase ในรูปเดียวกันนี้จะดีกว่าตอนที่ 1 เพราะว่าเมื่อเขียนตอนที่ 1 นั้นผู้เขียนลืมกำหนดรหัสคั่นระเบียนข้อมูลตามวิธีของดีเบสไป จึงได้เพิ่มให้ในตอนที่ 2 นี้ คือคำสั่งที่เขียนว่า DataDbase [(R-1)* SizeOfRecord]:= #32; และตัดส่วนเกินคือ WidthOfField[0]; ออกไปด้วย

และแก้ไขคำสั่ง readln มาเรียกใช้ชุดคำสั่ง InputSTR แทน ซึ่งจะทำให้การป้อนข้อความได้ตรงกับรูปแบบของแต่ละเขตข้อมูล ส่วนที่นอกจากกล่าวนี้ยังคงรูปแบบของคำสั่งเดิมของตอนที่ 1



```
PROCEDURE AppendDatabase;
var no,s,x,xx,R:word;  row,k:byte;
begin
   for no:=1 to 30 do FieldSTR[no]:='';
   clrscr;     writeln(' Append Database ');
   R:=CountOfRecord+1;  Head[4]:=R mod 256; Head[5]:=trunc(R/256);
   DisplayDatabase(R);  xx:=1;  row:=3;   CountOfRecord:=R;
   DataDbase[(R-1)*SizeOfRecord]:=#32;
 for No:=1 to CountOfField do
   begin
        Input_STR(15,row,WidthOfField[no],Fdec[no],No);
        x:=(R-1)*SizeOfRecord+xx;  k:=1;
        for s:=x to (x+WidthOfField[No]-1) do
         begin
           if k<=length(Inputstr) then DataDbase[s]:=InputStr[k]
                                  Else DataDbase[s]:=#32;
           inc(k);
         end;
        xx:=xx+WidthOfField[no];  inc(row);
   end;
end;

PROCEDURE EditDatabase(R:word);
var no,s,x,xx:word;  row,k:byte;
begin
Repeat
   clrscr;     writeln(' Edit Database ');
   DisplayDatabase(R);     xx:=1;  row:=3;
 for No:=1 to CountOfField do
   begin
     Input_STR(40,row,WidthOfField[no],Fdec[no],no);
     if InputSTR<>'' then
      begin
        x:=(R-1)*SizeOfRecord+xx;  k:=1;
        for s:=x to (x+WidthOfField[No]-1) do
         begin
           if k<=length(Inputstr) then DataDbase[s]:=InputStr[k]
                                  Else DataDbase[s]:=#32;
           inc(k);
         end;
      end;
        xx:=xx+WidthOfField[no];  inc(row);
   end;
 if R<CountOfRecord then begin
   writeln; write('    Do you want to edit another (Y/N) ');
   repeat ch:=readkey; until upcase(ch) in ['N','Y'];
   writeln;
   if upcase(ch)='N' then R:=CountOfRecord;
                     end;
    Inc(R);
until R>CountOfRecord;
end;
```

รูปที่ 4 ชุดคำสั่งในการเพิ่มข้อมูลและการแก้ไขข้อมูล

```
PROCEDURE FindField(x,y:byte);
var xx:word; No:byte;
begin
  No:=1;  cursorOff;
  Inverse(x,y,NameOfField[1]);
  Inverse(x+10,y,' : Move use key '#24#25'  , Select'#26'End key ');
   Repeat
     readchr;
     case Funckey of
      #72:if No>1 then dec(No) else No:=CountOfField;
      #80:if No<CountOfField then Inc(No) Else No:=1;
     end;
     normal(x,y,copy(Space,1,10));
     Inverse(x,y,NameOfField[no]);
   until Funckey=#79;  NoField:=No;
  xx:=1; No:=1;
   repeat
      If NoField=No then  No:=CountOfField+1;
      if No<=CountOfField then xx:=xx+WidthOfField[no];
      inc(no);
   until No>CountOfField;  Xstart:=xx;  CursorOn;
end;
```

รูปที่ 5 ชุดคำสั่งตรวจหาลำดับที่ของเขตข้อมูล

การหาลำดับที่ของเขตข้อมู

เมื่อเราประสงค์จะทำการแก้ไ
ข้อมูล, ค้นหาข้อมูล หรือ เปลี่ยนแปล
ข้อมูลนั้น สิ่งแรกที่ต้องทำก็คือการป้อ
ชื่อของเขตข้อมูล ซึ่งถ้าให้ป้อนผ่านแป้
พิมพ์ก็คงจะเป็นการยากเพราะอาจ
ป้อนชื่อไม่ตรงกับที่มีอยู่ในโครงสร้
ของแฟ้มข้อมูลก็ได้ ซึ่งจะทำให้ยุ่งย
ที่ต้องเขียนคำสั่งตรวจสอบอีก

เพื่อให้การป้อนชื่อเขตข้อมูลเ
ไปอย่างถูกต้องมากขึ้น จึงได้เขียน
คำสั่งขึ้นมาช่วยงาน คือชุดคำสั่ง Fin
Field ในรูปที่ 5 ซึ่งมารายละเอียดดั

กำหนดตัวแปร x,y เข้ามาเพื่อ
แสดงชื่อเขตข้อมูล ณ ตำแหน่งที่กำห
พร้อมกับแสดงให้ทราบว่าใช้แป้นลูก
ขึ้นลง สำหรับเลื่อนชื่อเขตข้อมูล เ
ให้กดแป้น End เมื่อต้องการเลือก
เขตข้อมูลที่ปรากฏในแถบสว่าง เมื่อ
การวนรอบ repeat นี้จะได้ตัวแปร
Field เก็บหมายเลขของเขตข้อมูล

เมื่อกดแป้น End ก็จะมาทำ
วนรอบ repeat อีกครั้งหนึ่ง เพื่อห
หาจุดเริ่มต้นของเขตข้อมูลที่เลือกไว้
(ตัวแปร Xstart)

การเปลี่ยนแปลงข้อมูล

การเปลี่ยนแปลงข้อมูลที่อยู่ใน
ข้อมูลใด ๆ ให้ไปเป็นข้อมูลอื่น สำห
กรณีที่ข้อมูลเหมือนกันหลายระเบ
ข้อมูลนั้น การใช้คำสั่งแก้ไขทีละระเบ
ข้อมูลจะเสียเวลามาก จึงต้องเขียน
คำสั่งให้เปลี่ยนข้อมูลแบบค้นหาข้อค
ที่ต้องการแล้วให้เปลี่ยนไปเป็นข้อมูล
ตั้งแต่ต้นระเบียนแรกจนถึงระเบ
สุดท้าย

ชุดคำสั่ง ChangeDatabase ใ

ที่ 6 จะช่วยดำเนินการในเรื่องนี้โดยมีรายละเอียดดังนี้

ส่งข้อความของเขตข้อมูลเดิม (OldField) และที่เปลี่ยนแปลงใหม่ (NewField) เข้ามาใช้งานในชุดคำสั่งนี้ โดยก่อนเรียกใช้ชุดคำสั่งนี้ได้ทำการเลือกเขตข้อมูลที่ต้องการเปลี่ยนมาก่อนแล้ว

ตรวจสอบระเบียนข้อมูลตั้งแต่ต้นจนจบแฟ้มข้อมูล (ช่วง repeat until NoR>CountOfRecord;) โดยทำการตรวจข้อความที่บรรจุอยู่ในเขตข้อมูลของระเบียนข้อมูลที่ตรวจสอบ (NoR) ว่าตรงกับ OldField หรือไม่ ถ้าชนิดของเขตข้อมูลที่จะเปลี่ยนนั้นเป็นชนิดตัวเลข (numeric:=true) จะตรวจสอบจากตัวแปรที่เป็นตัวเลขคือ NumFind:=NumField แทน

เมื่อตรวจสอบพบว่าตรงกันก็จะกำหนดตัวแปร ChangeOn:=true;

ถ้าตัวแปร ChangeOn เป็นจริงจะทำการเปลี่ยนข้อมูลของเขตข้อมูลนั้น ๆ ให้ตาม NewField และกำหนดให้ตัวแปร Change เป็นจริง และแสดงให้ทราบว่าข้อมูลเปลี่ยนไปที่ระเบียนข้อมูล

เมื่อวนรอบจนได้ค่า NoR>CountOfRecord ก็จบการแทนที่ข้อมูล ถ้าค่าตัวแปร Change เป็นเท็จ ก็ให้แสดงข้อความว่าไม่พบข้อมูลที่จะทำการเปลี่ยน

```
PROCEDURE ChangeDatabase(OldField,NewField:string);
var no,s,x,xx:word;  k:byte; ck:integer;
    Change:boolean;  STT:string;
    NumFind:real;    ChangeOn:Boolean;
begin
     clrscr;   writeln(' Change Database '); writeln;
     NoR:=1;   No:=1;   Change:=false;
 repeat
     x:=(NoR-1)*SizeOfRecord+XStart;
     S:=x;  ck:=0;  STT:='';  ChangeOn:=False;
    repeat
      STT:=STT+DataDbase[s];
      Inc(s); if DataDbase[s]=#0 then ck:=27;
      if (STT<>'') and (DataDbase[s]=#32) then ck:=27;
    until (s>(x+WidthOfField[NoField]-1)) or (ck=27);
   if Numeric then begin Val(STT,NumFind,ck);
                         if NumFind=NumField then ChangeOn:=true;
                   end
    else
     if copy(STT,1,length(OldField))=OldField then  ChangeOn:=true;
    if ChangeOn then
       begin  Change:=True;
          gotoxy(2,3); writeln('Change ',No,' records'); Inc(no);
          x:=(NoR-1)*SizeOfRecord+Xstart;  k:=1;
          for s:=x to (x+WidthOfField[NoField]-1) do
            begin
             if k<=length(NewField) then DataDbase[s]:=NewField[k]
                                    Else DataDbase[s]:=#32;
             inc(k);
            end;
       end;
     Inc(NoR);
 until NoR>CountOfRecord;
   if Pred(Change) then
       begin STT:='';
          for x:=1 to 10 do
            if NameOfField[noField][x] in [#33..#126] then
               STT:=STT+NameOfField[noField][x];
          writeln('=======================================');
          writeln('  FIELD : ',STT,'  not found  ',OldField);
          writeln('=======================================');
       end;
    ch:=readkey;
end;
```

รูปที่ 6 ชุดคำสั่งเปลี่ยนแปลงข้อมูล

การค้นหาข้อมูล

การค้นหาข้อมูลตามความต้องการของเราทำได้ดังชุดคำสั่งในรูปที่ 7 คือ SearchDatabase ซึ่งขออธิบายคร่าว ๆ ดังนี้

ก่อนจะเรียกใช้คำสั่งนี้จะต้องเลือกเขตข้อมูลมาก่อน

ในส่วนการวนรอบ repeat จะทำการค้นหาข้อความในเขตข้อมูลที่มีค่าตรงกับตัวแปร Condition ซึ่งเป็นข้อความที่ต้องการค้นหา โดยมีลักษณะการทำงานคล้ายกับชุดคำสั่ง Change Database โดยจะตรวจทั้งกรณีที่ชนิดของเขตข้อมูลเป็นตัวเลขด้วย ถ้าพบก็กำหนดให้ตัวแปร SearchOn เป็นจริง

เมื่อ SearchOn เป็นจริงก็ให้แสดงระเบียนข้อมูลที่ค้นพบนั้นบนจอภาพ และกำหนดให้ตัวแปรอีกตัวหนึ่งคือ Search เป็นจริง

เมื่อค้นจนถึงระเบียนสุดท้ายแล้ว หากตัวแปร Search เป็นเท็จก็แสดงให้ทราบว่าไม่พบข้อมูลที่ให้ต้องการ ก็เป็นการจบส่วนของการค้นหาข้อมูล

ในรูปที่ 7 นี้ยังมีชุดคำสั่ง Display สำหรับแสดงข้อมูลทั้งหมด (ตอนที่ 1 ไม่ได้เขียนเป็นชุดคำสั่ง) โดยจะทำงานได้ 2 กรณีคือ แสดงตั้งแต่ระเบียนแรกจนถึงระเบียนสุดท้ายวิธีหนึ่ง กับการแสดงแบบเลือกกลุ่มระเบียนข้อมูล (ในดีเบสคือ disp next) อีกแบบหนึ่ง

ขอให้ทำความเข้าใจกับการทำงานด้วยเช่นกัน

```
PROCEDURE SearchDatabase(condition:string);
var no,s,x:word;   ck:integer;
    Search,SearchOn:boolean;   STT:string;
    NumFind:real;
begin
     clrscr;   writeln(' Search Database '); writeln;
     Search:=false;    NoR:=1;
 repeat
     x:=(NoR-1)*SizeOfRecord+Xstart; STT:='';
     S:=x;  ck:=0;  SearchOn:=False;
       repeat
         STT:=STT+DataDbase[s];
         Inc(s); if DataDbase[s]=#0 then ck:=27;
       until (s>(x+WidthOfField[NoField]-1)) or (ck=27);
   if Numeric then begin Val(STT,NumFind,ck);
                         if NumFind=NumField then SearchOn:=true;
                   end
    else
     if copy(STT,1,length(Condition))=Condition then  SearchOn:=true;
      if SearchOn then
        begin  DisplayDatabase(NoR);
               Search:=true;  readchr;
        end;
     Inc(NoR);
  until NoR>CountOfRecord;
    if pred(Search) then
       begin   STT:='';
          for x:=1 to 10 do
            if NameOfField[noField][x] in [#33..#126] then
              STT:=STT+NameOfField[noField][x];
         writeln('=======================================');
         writeln('  FIELD : ',STT,'  not found  ',Condition);
         writeln('=======================================');
       end;
    ch:=readkey;
end;

PROCEDURE Display(StartR,EndR:word);
begin
    NoR:=StartR;
   repeat
       clrscr;
       writeln(' Display Database');
       DisplayDatabase(NoR);  inc(NoR);
       readchr; if ch=#27 then NoR:=EndR+1;
   until NoR>EndR;
end;
```

รูปที่ 7 ชุดคำสั่งค้นหาข้อมูลและการแสดงข้อมูลบางส่วน

เมนูหลักแบบเลื่อนแถบสว่าง

เพื่อให้การเลือกการทำงานเป็นไปด้วยความสะดวกขึ้น จึงขอเสนอเมนูหลักแบบเลื่อนแถบสว่างให้เป็นแนวทางหนึ่งแบบ ดังโปรแกรมในรูปที่ 8. (ซึ่งค่อนจะยาวนิดหนึ่ง) โดยจะคงใช้เมนูนี้ตลอดไปในตอนอื่นด้วย เพียงแต่จะกรณีทำงานเพิ่มขึ้นเท่านั้น มีรายละเอียดดังนี้

กำหนดชื่อกรณีทำงานไว้ 10 กรณี โดยให้กรณีสุดท้ายเป็นทางออกจากเมนู ซึ่งในตอนต่อ ๆ ไปก็เพียงเพิ่มบางส่วนตามงานที่ต้องการ โดยตัวแปรที่ต้องเปลี่ยนไปด้วยคือ Maxmenu ส่วนตัวแปร RowInc สำหรับกำหนดความห่างของระยะบรรทัดของรายการที่เลือก

โดยตัวแปร Choice เป็นตัวแปรสำหรับเลือกกรณีทำงาน

มีตัวแปรบูลีนที่ใช้งานคือ Dbf Change, Sort กำหนดไว้เพื่อกำหนดการแก้ไขข้อมูลและการเรียงลำดับ เป็นการกำหนดล่วงหน้าไว้ให้ผู้อ่านพัฒนาบางส่วนของชุดคำสั่งเพิ่มเติม คือการออกจากเมนูควรจะตรวจว่าได้มีการเก็บแฟ้มข้อมูลที่ได้ทำการแก้ไขแล้วหรือยัง เพื่อจะได้จัดการเก็บแฟ้มให้เรียบร้อย ต่อไป

ตอนนี้ขอให้ศึกษาชุดคำสั่งในรูป ที่ 8 ก่อน

การจัดรายการแสดงบนจอภาพนั้น ผู้เขียนได้จัดให้เรียงลำดับรายการเรียง จากซ้ายไปขวาก่อน แล้วเลื่อนลงมาที่ แถวข้างล่าง นั่นคือรายการที่ 1, 3, 5, 7, 9 จะเรียงอยู่ทางคอลัมน์ซ้ายมือ ส่วน รายการที่ 2, 4, 6, 8, 10 จะเรียงจาก บนลงล่างทางคอลัมน์ขวามือ ทำให้ด้าน ซ้ายมือเป็นกลุ่ม Choice mod 2 มีค่า เท่ากับ 1 ส่วนทางกลุ่มขวามือมีค่า Choice mod 2=0 นำมาใช้กำหนดการ เลื่อนแถบสว่างเมื่อกดแป้นลูกศรซ้าย ลูกศรขวา (ค่าตัวแปร funckey = #75, #77 ตามลำดับ) ได้

ส่วนแถวบนสุดจะมีค่าน้อยกว่า 3 (แถวที่สองมีค่ามากกว่า 2) นำมาใช้ ตรวจสอบการกดแป้นลูกศรขึ้น (ค่าตัว แปร funckey = #72) ส่วนรายการ เมนูที่อยู่ถัดจากแถวล่างขึ้นมาข้างบนมี ค่า 7, 8 จึงมีค่าน้อยว่าค่า Maxmenu-1 (ในที่นี้คือ 9) นำมาใช้ตรวจการกดแป้น ลูกศรลง (ค่าตัวแปร funckey = #80) เพื่อให้เลื่อนแถบตามต้องการได้ และ เลือกรายการด้วยการกดแป้น Enter ใน กรณีที่แถบสว่างอยู่ตรงกับรายการที่ ต้องการ

ในช่วงที่มีการเลื่อนแถบสว่างเลือก รายการนี้ได้กำหนดไม่ให้แสดงเคอร์เซอร์

มาดูการทำงานเมื่อเลือกรายการ ต่าง ๆ ให้ทำงาน (ใช้ตัวแปร Choice มาตรวจสอบ และเงื่อนไขรายการเหล่านี้ จะคงเดิมไปตลอดบทความชุดนี้ แต่อาจ มีรายการเพิ่มเติมสำหรับการทำงานอื่น ของแต่ละตอน

เมื่อเลือกรายการเรียกแฟ้มข้อมูล (UsedBASEFile เมื่อตัวแปร Choice=

1) จะให้ป้อนชื่อแฟ้มข้อมูลซึ่งสามารถป้อนได้เหมือนกับอยู่ในดอสเช่น c:\dBase\test.dbf หรือ b:test.dbf โดยจะป้อนสกุลหรือไม่ก็ได้เพราะมีคำสั่งตรวจสอบแล้ว จาก pos ('.',dBaseFile) และจะทำการเปลี่ยนตัวอักขระทั้งหมดที่ป้อนให้เป็นตัวใหญ่ทั้งหมด ทั้งนี้เพื่อจะสะดวกในการตรวจชื่อไฟล์ตอนที่จะเก็บ

จากนั้นทำการอ่านแฟ้มข้อมูลที่กำหนดชื่อเข้ามา ถ้าอ่านแฟ้มได้ก็ให้ตัวแปร UseDbf เป็นจริง และให้ตัวแปร DbfChange เป็นเท็จ

การเลือกรายการเก็บแฟ้มข้อมูล (SavedBASEFile, Choice=2) ให้ป้อนชื่อแฟ้มที่จะเก็บโดยถ้ากดแป้น Enter อย่างเดียวจะเก็บลงในชื่อเดิมที่เรียกเข้ามา ยกเว้นว่าได้ทำการเรียงลำดับข้อมูลใหม่(ตัวแปร Sort เป็นจริง) จะต้องป้อนชื่อใหม่ที่ไม่ซ้ำกับชื่อเดิม

เลือกรายการที่ 3 ก็จะแสดงโครงสร้างของแฟ้มข้อมูล

เลือกรายการที่ 4 จะแสดงข้อมูลตั้งแต่ระเบียนที่ 1 ไปจนถึงระเบียนสุดท้าย โดยถ้ากดแป้น ESC ก็จะออกจากส่วนนี้กลับมายังเมนูทันทีโดยไม่แสดงจนถึงระเบียนสุดท้าย

เลือกรายการที่ 5 จะให้ป้อนหมายเลขระเบียนที่จะเริ่มต้นให้แสดงโดยจะป้อนได้ในช่วงของข้อมูลที่มีอยู่เท่านั้น (1..CountOfRecord) ตามด้วยการป้อนจำนวนระเบียนข้อมูลที่ต้องการให้แสดง เมื่อป้อนครบก็จะแสดงข้อมูลให้แบบเดียวกับรายการที่ 4 เพียงแต่จำนวนระเบียนข้อมูลจะน้อยกว่าเท่านั้น

เลือกรายการที่ 6 เป็นการเพิ่มข้อมูลซึ่งจะเพิ่มได้เรื่อย ๆ จนกว่าจะกดแป้น N เมื่อมีคำถามท้ายการเพิ่มข้อมูล ซึ่งจะทำให้ตัวแปร DbfChange

รูปที่ 8 ชุดคำสั่งเมนูหลักแบบเลื่อนแถบสว่างเพื่อเลือกกรณีทำงาน

```
PROCEDURE Menu;
const MenuStr:array[1..10] of string[20]=
        (' UsedBASEFile ',' SavedBASEFile ',' ReadStructure ',
         ' DisplayAll ',' DisplayNext ',' Appenddata ',' Editdata ',
         ' Searchdata ',' Changedata ',' Quit ');
var choice,Row,MaxMenu,RowInc:byte;      ck:integer;
    Search,OldField,NewField:string; StartR,Next:word;
    DbfChange,Sort:boolean;

begin
  UseDbf:=false;  DbaseFile:='';  Edit:=false;    Sort:=false;
  MaxMenu:=10;    RowInc:=2;      Choice:=1;  Row:=5;
 repeat
  clrscr;  Next:=5;  normal(2,3,'File : '+DbaseFile);
  Inverse(11,1,' Management dBASE by Pascal  *** Arthan Sattayarak *** ');
  Inverse(18,24,'  Move use key '#27#24#25#26'  , Select'#26'Enter key ');
  for Ck:=1 to Maxmenu do
    begin
      if ck mod 2 = 1 then Normal(15,Next,MenuStr[Ck])
        Else begin Normal(47,Next,MenuStr[Ck]); Inc(next,RowInc); end;
    end;
     CursorOff; Inverse(15+32*(1-Choice mod 2),row,MenuStr[Choice]);
  repeat
     readchr;
    case funckey of
     #72:begin  if Choice>2 then   begin
                Normal(15+32*(1-Choice mod 2),row,MenuStr[Choice]);
                dec(Choice,2);  dec(row,RowInc);
                Inverse(15+32*(1-Choice mod 2),row,MenuStr[Choice]);
         end;                          end;
     #80:begin   if Choice<Maxmenu-1 then   begin
                Normal(15+32*(1-Choice mod 2),row,MenuStr[Choice]);
                Inc(Choice,2); Inc(row,RowInc);
                Inverse(15+32*(1-Choice mod 2),row,MenuStr[Choice]);
         end;                                 end;
     #75:begin if Choice mod 2=0 then  begin
                Normal(15+32*(1-Choice mod 2),row,MenuStr[Choice]);
                dec(Choice);
                Inverse(15+32*(1-Choice mod 2),row,MenuStr[Choice]);
         end;                                  end;
     #77:begin if Choice mod 2=1 then begin
                Normal(15+32*(1-Choice mod 2),row,MenuStr[Choice]);
                Inc(Choice);
                Inverse(15+32*(1-Choice mod 2),row,MenuStr[Choice]);
         end;                                 end;
   end;
   if (Choice in [3..Maxmenu-1]) and pred(UseDbf) and (ch=#13) then Ch:=#0;
  until ch=#13;
    CursorOn; Normal(18,24,Space); gotoxy(1,20); Numeric:=False;
    case Choice of
     1:begin  write('     File : ');  readln(DbaseFile);
              if pos('.',DbaseFile)=0 then DbaseFile:=DbaseFile+'.DBF';
              for ck:=1 to length(DbaseFile) do
                    Dbasefile[ck]:=Upcase(DbaseFile[ck]);
              UseDbf:=true; Read_dataBase;
              if pred(UseDbf) then DbaseFile:='' Else DbfChange:=False;
       end;

     2:begin repeat  Normal(16,22,space);
              gotoxy(1,22); write('    New Name : ');  readln(InputSTR);
              for ck:=1 to length(InputSTR) do
                      InputSTR[ck]:=Upcase(InputSTR[ck]); ck:=0;
              if (pos('.',InputSTR)=0) and (InputSTR<>'') then
                                      InputSTR:=InputSTR+'.DBF';
              if Sort then
                 if (InputSTR=DbaseFile) or (InputSTR='') then ck:=27;
             until ck=0;
              if InputSTR<>'' then DBaseFile:=InputSTR;
              SaveDatabase(DbaseFile);
       end;
     3:begin clrscr; ReadStructure; end;
     4:DisPlay(1,CountOfRecord);
     5:begin
        repeat
          gotoxy(31,22); writeln(copy(Space,1,6)+'(1-',CountOfRecord,')');
          gotoxy(9,22); write('Record no. to start : '); readln(InputSTR);
          val(InputSTR,StartR,ck);
        until StartR in [1..CountOfRecord];
        write('   CountofRecord to display : ');  readln(InputSTR);
        val(InputSTR,Next,ck);   Next:=StartR+Next; dec(next);
        if Next>CountOfRecord then Next:=CountOfRecord;
        Display(StartR,Next);
       end;
     6:begin  repeat  AppendDatabase;
```

รูปที่ 8 (ต่อ)

```
            writeln; write('    Do you want to Append another (Y/N) ');
                repeat  readchr;
                until upcase(ch) in ['N','Y'];
            until Upcase(ch)='N'; DbfChange:=True;
      end;
    7:begin
       repeat
        gotoxy(21,22); writeln(copy(Space,1,6)+'(1-',CountOfRecord,')');
        gotoxy(4,22);  write('Edit record no : '); readln(InputSTR);
        val(InputSTR,NoR,ck);
       until NoR in [1..CountOfRecord];
        Edit:=true; EditDatabase(NoR); EDit:=False; DbfChange:=True;
      end;
    8:begin  Normal(5,20,'Field name : ');  FindField(18,20);
             Normal(5,21,'Search for : ');
        Input_STR(18,21,WidthOfField[NoField],Fdec[NoField],NoField);
        Search:=InputSTR; if Search='' then Search:=' ';
           If TypeOfField[NoField]='N' then
                begin Numeric:=true;
                      Val(Search,NumField,ck);
                end;
        SearchDatabase(Search);
      end;
    9:begin  normal(5,20,'Field name : ');  FindField(18,20);
             normal(4,21,'Change from : ');
       Input_STR(18,21,WidthOfField[NoField],Fdec[NoField],NoField);
       OldField:=InputSTR; if OldField='' then  OldField:=' ';
           If TypeOfField[NoField]='N' then
                begin Numeric:=true;
                      Val(OldField,NumField,ck);
                end;
        normal(13,22,'to : ');
        Input_STR(18,22,WidthOfField[NoField],Fdec[NoField],NoField);
        NewField:=InputSTR; if NewField='' then  NewField:=' ';
        ChangeDatabase(OldField,NewField);   DbfChange:=True;
      end;
   end;
 until Choice>=MaxMenu;
end;

{================= Read dBase_files with pascal (II) ================}

begin
   Fillchar(Space,41,#32); Space[0]:=#40;
   Menu;    CursorOff;
   Inverse(32,24,' End of Program ');
   readchr; CursorOn;
end.
```

เป็นจริง

เลือกรายการที่ 7 เป็นการแก้ไขข้อมูลจะให้ป้อนหมายเลขของระเบียนข้อมูลที่ต้องการแก้ไขก่อนถึงไปยังการแก้ไขข้อมูล ซึ่งเมื่อแก้ไขเสร็จสิ้นแล้วจะมีคำถามว่าจะทำการแก้ไขระเบียนอื่นหรือไม่ ถ้าแก้ไขก็จะแสดงรายการของระเบียนที่อยู่ถัดไปให้ ถ้าเลือกไม่แก้ไขก็จะกลับไปยังเมนู โดยตัวแปร Dbf Change จะเป็นจริงหลังการแก้ไขข้อมูลสิ้นสุด

เลือกรายการที่ 8 เป็นการค้นหาข้อมูล จะให้เลือกชื่อเขตข้อมูลก่อนแล้วป้อนข้อมูลที่จะค้นหาโดยจะป้อนข้อมูลได้ตามรูปแบบของข้อมูลที่ถูกกำหนดในเขตข้อมูลนั้น ๆ ถ้าชนิดของเขตข้อมูลเป็นแบบตัวเลขก็จะเปลี่ยนค่าที่ป้อนนี้ไปเป็นตัวเลข เพื่อค้นหาจากชุดคำสั่ง SearchDatabase ต่อไป

เลือกรายการที่ 9 เป็นการเปลี่ยนข้อมูลเดิมให้เป็นข้อมูลใหม่ เริ่มด้วยเลือกชื่อเขตข้อมูลตามด้วยการป้อนข้อมูลเดิมและข้อมูลใหม่โดยจะตามรูปแบบที่กำหนดไว้ในเขตข้อมูลนั้น แล้วเรียกใช้ชุดคำสั่งการเปลี่ยนแปลงข้อมูลหรือชุดคำสั่ง ChangeDatabase มาทำงาน

เลือกรายการที่ 10 ก็จะออกจากเมนูและจบการทำงาน ในตอนต่อไปรายการเลิกงานนี้จะเลื่อนไปอยู่รายการสุดท้าย (MaxMenu) ทุกครั้ง คือเอารายการอื่น ๆ ที่เขียนใหม่เข้ามาแทรกก่อน

ส่วนท้ายสุดหรือส่วนโปรแกรมหลักนั้น เริ่มด้วยการกำหนดให้ตัวแปร space ประกอบด้วยรหัสแอสกี 32 จำนวน 41 ตัวและกำหนดความยาวของตัวแปรนี้เป็น 40 ไบต์ แล้วเรียกชุดคำสั่งเมนูมาทำงานโดยจบลงด้วยการแสดงข้อความให้รู้ว่าเป็นการจบโปรแกรม

คงต้องจบตอนที่สองลงเพียงแค่นี้ก่อนเพราะค่อนข้างจะยาวมาก หวังว่าผู้อ่านทุกท่านคงได้ประโยชน์จากโปรแกรมที่เสนอนี้ไม่มากก็น้อย พบกันใหม่ฉบับหน้า...